



UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/601,015	06/19/2003	Eric Teh Gim Aik	ALTRP085/ A880	7118
51501	7590	05/05/2006	EXAMINER	
BEYER WEAVER & THOMAS, LLP			ROSSOSHEK, YELENA	
ATTN: ALTERA			ART UNIT	
P.O. BOX 70250			PAPER NUMBER	
OAKLAND, CA 94612-0250			2825	

DATE MAILED: 05/05/2006

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary

Application No.

10/601,015

Applicant(s)

AIK, ERIC TEH GIM

Examiner

Helen Rossoshek

Art Unit

2825

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

- 1) ☒ Responsive to communication(s) filed on 14 February 2006.
- 2a) ☒ This action is **FINAL**. 2b) ☐ This action is non-final.
- 3) ☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

- 4) ☒ Claim(s) 1-3, 5-17, 19-34, 35-49 is/are pending in the application.
- 4a) Of the above claim(s) 35-49 is/are withdrawn from consideration.
- 5) ☐ Claim(s) _____ is/are allowed.
- 6) ☒ Claim(s) 1-3, 5-17 and 19-34 is/are rejected.
- 7) ☐ Claim(s) _____ is/are objected to.
- 8) ☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

- 9) ☐ The specification is objected to by the Examiner.
- 10) ☐ The drawing(s) filed on _____ is/are: a) ☐ accepted or b) ☐ objected to by the Examiner.
Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).
Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).
- 11) ☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

- 12) ☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).
- a) ☐ All b) ☐ Some * c) ☐ None of:
1. ☐ Certified copies of the priority documents have been received.
 2. ☐ Certified copies of the priority documents have been received in Application No. _____.
 3. ☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

- | | |
|--|---|
| 1) <input type="checkbox"/> Notice of References Cited (PTO-892) | 4) <input type="checkbox"/> Interview Summary (PTO-413)
Paper No(s)/Mail Date. _____ |
| 2) <input type="checkbox"/> Notice of Draftsperson's Patent Drawing Review (PTO-948) | 5) <input type="checkbox"/> Notice of Informal Patent Application (PTO-152) |
| 3) <input type="checkbox"/> Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08)
Paper No(s)/Mail Date _____ | 6) <input type="checkbox"/> Other: _____ |

DETAILED ACTION

1. This office action is in response to the Application 10/601,015 filed 06/19/2003 and amendment filed 02/14/2006.

2. Claims 1-3, 5-17, 19-34 and 35-49 remain pending in the Application. Claims 1-3, 5-17, 19-34 as a Group I is selected with traverse in response to the restriction. Claims 35-49 have been withdrawn from consideration.

3. Applicant's election of Group I (claims 1-3, 5-17, 19-34) for prosecution with traverse is acknowledged. The grounds for traverse are: weighting a burden if restriction is not required after the first action on the merits; and the search and examination of the entire Application could be made without serious burden.

The traverse has been fully considered and found not persuasive, because MPEP 811 states:

37 CFR 1.142(a), second sentence, **>indicates that a restriction requirement "will normally< be made before any action upon the merits; however, it may be made at any time before final action **." This means the examiner should make a proper requirement as early as possible in the prosecution, in the first action if possible, otherwise, as soon as the need for a proper requirement develops.

Moreover, Group I and Group II directed to the patentably distinct species of the claimed invention, such as applying set of predefined rules to the information of the plurality of nodes in the netlist and determine whether **any of the predefined rules** have been **violated by any of the plurality of nodes** (Group I) and **at the first node** (not plurality of nodes) of the netlist employing a generic routine to execute a first design rule and then second design rule and determine whether properties of the **first node** violate the first and second design rules (Group II). Therefore, Group II does not

consider going to the next node and/or **any plurality of nodes**.

Because these inventions are distinct for the reasons given above, and the search for one group is not required for another group, restriction for examination purposes as indicated is proper.

The restriction requirement has been fully reconsidered, and **is still deemed proper and therefore, made FINAL**. The elected claims 1-3, 5-17 and 19-34 will be examined in this office action. Claims 35-49 are withdrawn from consideration. The Applicant is advised, that cancellation of non-elected claims is required.

Claim Rejections - 35 USC § 102

7. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

A person shall be entitled to a patent unless –

(b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

8. Claims 1-3, 5-17, 9-34 are rejected under 35 U.S.C. 102(b) as being anticipated by Fairbanks (US Patent 6,182,020).

With respect to claims 1, 31 Fairbanks teaches a method (col. 1, ll.65-67), apparatus and program for performing design rule checking on an electronic design within software base system such as MAX-PLUS available from Altera including computer design tools as software-implemented tool for assisting in the design of logic circuits providing technique for checking a user's design against a set of the electronic design against a set of design rules (col. 1, ll.17-20, ll.21-26); the method comprising: receiving a netlist of the electronic design, the netlist having a plurality of nodes as

Art Unit: 2825

shown on the Fig. 1, wherein the netlist is stored in the file 10 and received by converting into a simulator logic netlist 14 (col. 1, ll.30-33; col. 3, ll.46-49); extracting information from each of the plurality of nodes in the netlist, wherein the extracting information operation is done only once for each node in the netlist, whereby the extracted information is available to a set of predefined rules within simulator netlist extraction (SNF) wherein the subset of design rules is available for each device (node) in the extracted nodes of the netlist and executed to check a violation of the rules (col. 4, ll.7-11) also as shown in a portion of the program code (col.7, ll.42-45) wherein the extracted information for each device D (node) in netlist N is demonstrated, wherein extracted information from each of the plurality of nodes is done once and stored in the database including node id with a pointer for location of the extracted information of each node (col. 19, ll.58-67; col. 17, ll.7-10); applying the set of predefined rules to the previously extracted information of the plurality of nodes in the list as shown on the Fig. 6 (as a flow diagram of a system for designing and testing logic circuitry) within a step of applying a design rule to the logic design, wherein defining logic design is done by extracting the information from netlist as shown on the Fig. 1, also within an ability of the method to create a structure to define a node in the logic tree captured in the netlist (col. 19, ll.49-55); and determining whether any of the predefined rules have violated by any of the plurality of nodes, wherein the applying operation is performed by a rule checking routine, which checks the electronic design as shown on the Fig. 6 within a step "does the logic design violate the rule?" (col. 1, ll.65-67; col. 4, ll.39-40).

With respect to claim 5 Fairbanks teaches a method (col. 1, ll.65-67), the method comprising: receiving a netlist of the electronic design, the netlist having a plurality of nodes as shown on the Fig. 1, wherein the netlist is stored in the file 10 and received by converting into a simulator logic netlist 14 (col. 1, ll.30-33; col. 3, ll.46-49); extracting information from each of the plurality of nodes in the netlist, whereby the extracted information is available to a set of predefined rules, wherein the extracting information operation comprises determining what information is required to apply the predefined rules to the plurality of nodes, within simulator netlist extraction (SNF) wherein the subset of design rules is available for each device (node) in the extracted nodes of the netlist and executed to check a violation of the rules (col. 4, ll.7-11) also as shown in a portion of the program code (col.7, ll.42-45) wherein the extracted information for each device D (node) in netlist N is demonstrated, wherein extracted information from each of the plurality of nodes is done and stored in the database including node id with a pointer for location of the extracted information of each node (col. 19, ll.58-67; col. 17, ll.7-10), wherein a routine drc_check_hazards has an ability to determine what parts of the database (storing extracted information for each node) to check by using a function drc_check_function_for_hazard (col. 10, 6-10, 16-19); applying the set of predefined rules to the previously extracted information of the plurality of nodes in the list as shown on the Fig. 6 (as a flow diagram of a system for designing and testing logic circuitry) within a step of applying a design rule to the logic design, wherein defining logic design is done by extracting the information from netlist as shown on the Fig. 1, also within an ability of the method to create a structure to define a node in the logic tree captured in

the netlist (col. 19, ll.49-55); and determining whether any of the predefined rules have violated by any of the plurality of nodes, wherein the applying operation is performed by a rule checking routine, which checks the electronic design as shown on the Fig. 6 within a step "does the logic design violate the rule?" (col. 1, ll.65-67; col. 4, ll.39-40).

With respect to claim 6 Fairbanks teaches a method (col. 1, ll.65-67), the method comprising: receiving a netlist of the electronic design, the netlist having a plurality of nodes as shown on the Fig. 1, wherein the netlist is stored in the file 10 and received by converting into a simulator logic netlist 14 (col. 1, ll.30-33; col. 3, ll.46-49); extracting information from each of the plurality of nodes in the netlist, wherein the extracted information comprises one or more properties selected from the group consisting of fanin count, fanout count, atom type, and port source atom type, whereby the extracted information is available to a set of predefined rules, within simulator netlist extraction (SNF) wherein the subset of design rules is available for each device (node) in the extracted nodes of the netlist and executed to check a violation of the rules (col. 4, ll.7-11) also as shown in a portion of the program code (col.7, ll.42-45) wherein the extracted information for each device D (node) in netlist N is demonstrated, wherein extracted information from each of the plurality of nodes is done and stored in the database including node id with a pointer for location of the extracted information of each node (col. 19, ll.58-67; col. 17, ll.7-10), wherein extracted information for each node stored in the database and includes the data, such input-counts, output_counts, atom type (node type) as a structure tree node (col. 19, 49-55; col. 20, ll.24-31); applying the set of predefined rules to the previously extracted information of the

plurality of nodes in the list as shown on the Fig. 6 (as a flow diagram of a system for designing and testing logic circuitry) within a step of applying a design rule to the logic design, wherein defining logic design is done by extracting the information from netlist as shown on the Fig. 1, also within an ability of the method to create a structure to define a node in the logic tree captured in the netlist (col. 19, ll.49-55); and determining whether any of the predefined rules have violated by any of the plurality of nodes, wherein the applying operation is performed by a rule checking routine, which checks the electronic design as shown on the Fig. 6 within a step “does the logic design violate the rule?” (col. 1, ll.65-67; col. 4, ll.39-40).

With respect to claim 11 Fairbanks teaches a method (col. 1, ll.65-67), the method comprising: receiving a netlist of the electronic design, the netlist having a plurality of nodes as shown on the Fig. 1, wherein the netlist is stored in the file 10 and received by converting into a simulator logic netlist 14 (col. 1, ll.30-33; col. 3, ll.46-49); extracting information from each of the plurality of nodes in the netlist, whereby the extracted information is available to a set of predefined rules, within simulator netlist extraction (SNF) wherein the subset of design rules is available for each device (node) in the extracted nodes of the netlist and executed to check a violation of the rules (col. 4, ll.7-11) also as shown in a portion of the program code (col.7, ll.42-45) wherein the extracted information for each device D (node) in netlist N is demonstrated, wherein extracted information from each of the plurality of nodes is done and stored in the database including node id with a pointer for location of the extracted information of each node (col. 19, ll.58-67; col. 17, ll.7-10); receiving the set pf predefined rules,

Art Unit: 2825

wherein each predefined rule conforms to a single specified format, wherein the format is a hierarchical arrangement comprising a rule -> a sub-rule -> a basic rule within the representing the hierarchy of levels of design rule compliance and the user-selectable list of optional rule selections in accordance with hierarchical arrangement of design rules (col. 4, ll.8-10; col. 2, ll.14-18; ll.22-25); applying the set of predefined rules to the previously extracted information of the plurality of nodes in the list as shown on the Fig. 6 (as a flow diagram of a system for designing and testing logic circuitry) within a step of applying a design rule to the logic design, wherein defining logic design is done by extracting the information from netlist as shown on the Fig. 1, also within an ability of the method to create a structure to define a node in the logic tree captured in the netlist (col. 19, ll.49-55); and determining whether any of the predefined rules have violated by any of the plurality of nodes, wherein the applying operation is performed by a rule checking routine, which checks the electronic design as shown on the Fig. 6 within a step "does the logic design violate the rule?" (col. 1, ll.65-67; col. 4, ll.39-40).

With respect to claim 16 Fairbanks teaches an apparatus for performing design rule checking on an electronic design apparatus and program for performing design rule checking on an electronic design within software base system such as MAX-PLUS available from Altera including computer design tools as software-implemented tool for assisting in the design of logic circuits providing technique for checking a user's design against a set of design rules (col. 1, ll.17-20, ll.21-26), the apparatus comprising: a netlist creator that produces a netlist describing the functional representation of the electronic design across a plurality of nodes as shown on the Fig. 1 typical computer

Art Unit: 2825

logic simulation technique proceeds within converting (creating) the netlist stored in a file 10 into a simulator logic netlist 14 describing functional representation of the electronic design a plurality of nodes (col. 1, ll.28-33); an information extractor that extracts information from the plurality of nodes in the netlist, the information being specific to each node, wherein the information extractor extracts information only once for each node in the netlist within simulator netlist extraction (SNF) wherein the subset of design rules is available for each device (node) in the extracted nodes of the netlist and executed to check a violation of the rules (col. 4, ll.7-11) also as shown in a portion of the program code (col.7, ll.42-45) wherein the extracted information for each device D (node) in netlist N is demonstrated, wherein extracted information from each of the plurality of nodes is done once and stored in the database including node id with a pointer for location of the extracted information of each node (col. 19, ll.58-67; col. 17, ll.7-10); a rule checking engine that applies a set of predefined rules to the extracted information of the plurality of nodes in the netlist and that identifies whether any of the predefined rules has been violated by any portion of the electronic design as shown on the Fig. 6 (as a flow diagram of a system for designing and testing logic circuitry) within a step of applying a design rule to the logic design, wherein defining logic design is done by extracting the information from netlist as shown on the Fig. 1, also within an ability of the method to create a structure to define a node in the logic tree captured in the netlist (col. 19, ll.49-55);

With respect to claim 19 Fairbanks teaches an apparatus for performing design rule checking on an electronic design apparatus and program for performing design rule

Art Unit: 2825

checking on an electronic design within software base system such as MAX-PLUS available from Altera including computer design tools as software-implemented tool for assisting in the design of logic circuits providing technique for checking a user's design against a set of design rules (col. 1, ll.17-20, ll.21-26), the apparatus comprising: a netlist creator that produces a netlist describing the functional representation of the electronic design across a plurality of nodes as shown on the Fig. 1 typical computer logic simulation technique proceeds within converting (creating) the netlist stored in a file 10 into a simulator logic netlist 14 describing functional representation of the electronic design a plurality of nodes (col. 1, ll.28-33); an information extractor that extracts information from the plurality of nodes in the netlist, the information being specific to each node, wherein the information extractor extracts information only once for each node in the netlist within simulator netlist extraction (SNF) wherein the subset of design rules is available for each device (node) in the extracted nodes of the netlist and executed to check a violation of the rules (col. 4, ll.7-11) also as shown in a portion of the program code (col.7, ll.42-45) wherein the extracted information for each device D (node) in netlist N is demonstrated, wherein extracted information from each of the plurality of nodes is done once and stored in the database including node id with a pointer for location of the extracted information of each node (col. 19, ll.58-67; col. 17, ll.7-10), wherein the information extractor extracts information based on what is required to apply a set of predefined rules to the plurality of nodes wherein extracted information from each of the plurality of nodes is done and stored in the database including node id with a pointer for location of the extracted information of each node (col. 19, ll.58-67;

Art Unit: 2825

col. 17, ll.7-10), and wherein a routine `drc_check_hazards` has an ability to determine what parts of the database (storing extracted information for each node) to check by using a function `drc_check_function_for_hazard` (col. 10, 6-10, 16-19); a rule checking engine that applies a set of predefined rules to the extracted information of the plurality of nodes in the netlist and that identifies whether any of the predefined rules has been violated by any portion of the electronic design as shown on the Fig. 6 (as a flow diagram of a system for designing and testing logic circuitry) within a step of applying a design rule to the logic design, wherein defining logic design is done by extracting the information from netlist as shown on the Fig. 1, also within an ability of the method to create a structure to define a node in the logic tree captured in the netlist (col. 19, ll.49-55).

With respect to claim 20 Fairbanks teaches an apparatus for performing design rule checking on an electronic design apparatus and program for performing design rule checking on an electronic design within software base system such as MAX-PLUS available from Altera including computer design tools as software-implemented tool for assisting in the design of logic circuits providing technique for checking a user's design against a set of design rules (col. 1, ll.17-20, ll.21-26), the apparatus comprising: a netlist creator that produces a netlist describing the functional representation of the electronic design across a plurality of nodes as shown on the Fig. 1 typical computer logic simulation technique proceeds within converting (creating) the netlist stored in a file 10 into a simulator logic netlist 14 describing functional representation of the electronic design a plurality of nodes (col. 1, ll.28-33); an information extractor that

Art Unit: 2825

extracts information from the plurality of nodes in the netlist, the information being specific to each node, wherein the information extractor extracts information only once for each node in the netlist within simulator netlist extraction (SNF) wherein the subset of design rules is available for each device (node) in the extracted nodes of the netlist and executed to check a violation of the rules (col. 4, ll.7-11) also as shown in a portion of the program code (col.7, ll.42-45) wherein the extracted information for each device D (node) in netlist N is demonstrated, wherein extracted information from each of the plurality of nodes is done once and stored in the database including node id with a pointer for location of the extracted information of each node (col. 19, ll.58-67; col. 17, ll.7-10), wherein the information extractor extracts information that is selected from the group consisting of fanin count, fanout count, atom type and port source atom type within extracted information for each node, stored in the database, which includes the data, such as input-counts, output_counts, atom type (node type) as a structure tree node (col. 19, 49-55; col. 20, ll.24-31); a rule checking engine that applies a set of predefined rules to the extracted information of the plurality of nodes in the netlist and that identifies whether any of the predefined rules has been violated by any portion of the electronic design as shown on the Fig. 6 (as a flow diagram of a system for designing and testing logic circuitry) within a step of applying a design rule to the logic design, wherein defining logic design is done by extracting the information from netlist as shown on the Fig. 1, also within an ability of the method to create a structure to define a node in the logic tree captured in the netlist (col. 19, ll.49-55).

With respect to claim 25 Fairbanks teaches an apparatus for performing design rule checking on an electronic design apparatus and program for performing design rule checking on an electronic design within software base system such as MAX-PLUS available from Altera including computer design tools as software-implemented tool for assisting in the design of logic circuits providing technique for checking a user's design against a set of design rules (col. 1, ll.17-20, ll.21-26), the apparatus comprising: a netlist creator that produces a netlist describing the functional representation of the electronic design across a plurality of nodes as shown on the Fig. 1 typical computer logic simulation technique proceeds within converting (creating) the netlist stored in a file 10 into a simulator logic netlist 14 describing functional representation of the electronic design a plurality of nodes (col. 1, ll.28-33); an information extractor that extracts information from the plurality of nodes in the netlist, the information being specific to each node, wherein the information extractor extracts information only once for each node in the netlist within simulator netlist extraction (SNF) wherein the subset of design rules is available for each device (node) in the extracted nodes of the netlist and executed to check a violation of the rules (col. 4, ll.7-11) also as shown in a portion of the program code (col.7, ll.42-45) wherein the extracted information for each device D (node) in netlist N is demonstrated, wherein extracted information from each of the plurality of nodes is done once and stored in the database including node id with a pointer for location of the extracted information of each node (col. 19, ll.58-67; col. 17, ll.7-10); a rule creator for preparing a set of predefined rules wherein the rule creator is configured for receiving the set of predefined rules from a user, and wherein each

Art Unit: 2825

predefined rule conforms to a single specified format wherein the specified format is a hierarchical arrangement of a rule -> a sub-rule -> a basic rule within a tool Design doctor, which creates different set of rules and sub-rules (subset) (col. 4, ll.8-10), which will be available and executed for selected extracted information of each node in the node tree and having an ability to customize any set of rules (col. 4, ll.13-15; 28-32) and within the representing the hierarchy of levels of design rule compliance and the user-selectable list of optional rule selections in accordance with hierarchical arrangement of design rules (col. 2, ll.14-18; ll.22-25); a rule checking engine that applies a set of predefined rules to the extracted information of the plurality of nodes in the netlist and that identifies whether any of the predefined rules has been violated by any portion of the electronic design as shown on the Fig. 6 (as a flow diagram of a system for designing and testing logic circuitry) within a step of applying a design rule to the logic design, wherein defining logic design is done by extracting the information from netlist as shown on the Fig. 1, also within an ability of the method to create a structure to define a node in the logic tree captured in the netlist (col. 19, ll.49-55); a traverser engine that recognizes whether any of the predefined rules require extracted information from a neighboring node and further identifies the neighboring node for evaluating with the rule checking engine if any of the predefined rules require extracted information from the neighboring node, the neighboring node being separate from a node currently under comparison within simulator netlist extraction (SNF) wherein the subset of design rules is available for each device (node) in the extracted nodes of the netlist and executed to check a violation of the rules (col. 4, ll.7-11) also as shown in a

Art Unit: 2825

portion of the program code (col.7, ll.42-45) wherein the extracted information for each device D (node) in netlist N is demonstrated, wherein extracted information from each of the plurality of nodes is done and stored in the database including node id with a pointer for location of the extracted information of each node (col. 19, ll.58-67; col. 17, ll.7-10), wherein the structure defines the node in the logic tree and lists the number of other nodes that are inputs into the node under checking (col. 19, ll.60-63).

With respect to claims 2, 3, 7-10, 12-15, 17, 21-24, 26-30, 32-34 Fairbanks teaches:

claims 2, 17, 32: the applying operation comprises: recognizing whether any of the predefined rules require extracted information from a neighboring node, the neighboring node being different from a node currently under consideration as shown in the program code on the column 17:

```
drc_check_macrofunctions( )  
    {  
        read in the device family for this compilation  
        read in the database of macrofunctions  
        for all nodes in the hierarchy tree  
            {  
                look up the record for this macrofunction  
                see if there are more efficient macrofunctions listed for  
                    this entry  
                if so, inform the user of the better possible choice  
            }  
    }
```

within a hierarchical structure of the node tree and checking this nodes in the hierarchical order by DRC; and identifying the neighboring node for comparing under the rule checking routine if any of the predefined rules require extracted information

from the neighboring node, wherein the recognizing and identifying operations are collectively performed by a traverser routine as shown above in the program code for design rule checking and looking for neighboring nodes in the hierarchical order checking all nodes going from one node to another (col. 17, ll.-28), wherein the structure defines the node in the logic tree and lists the number of other nodes that are inputs into the node under checking (col. 19, ll.60-63);

claim 3: the recited operations are each done automatically, without user intervention within software base system such as MAX-PLUS available from Altera including computer design tools as software-implemented tool for assisting in the design of logic circuits providing technique for checking a user's design against a set of design rules (col. 1, ll.17-20, ll.21-26);

claims 7, 21, 33: receiving the set of predefined rules (col. 3, ll.46-49);

claims 8, 22: the set of predefined rules comprises at least one hardcoded rule, which was predefined without any user intervention by default executing all rules to check logic design (col. 4, ll.33-34, ll.39-40);

claims 9, 23: the set of predefined rules comprises at least one hardcoded rule, which has been modified in accordance with user instructions within system's ability of user's customization of the set of rules (col. 4, ll.15; col. 3, ll.57-60);

claims 10, 24: each predefined rule conforms to a single specified format (col. 3, ll.57-60);

claims 12, 26: the applying the set of predefined rules to the previously extracted information of the plurality of nodes in the netlist comprises: applying a plurality of sub-

Art Unit: 2825

rules related by a logical operator (col. 4, ll.7-8) and as shown in the program code in the column 12

```

drc_create_sum_of_products(DRCS_TREE_NODE **node)
    }
    expand all exclusive-or (XOR) operators in the tree
    convert all NAND and NOR operators using DeMorgan's inversion
    on the terms and subtrees feeding a node
    convert all stacked gates, e.g. AND(AND( . . . )) to single gates
    new_node = drc_compass_tree(node);
    return (new_node);

```

claims 13, 27: the logical operator is selected from the group consisting of AND, NAND, NOR, XOR and OR as shown above in the example of program code on the column 12, (col. 19, ll.27-36);

claims 14, 28: the plurality of predefined rules is selected from the group consisting of electrical rules, connectivity rules, clock rules, timing closure rules, reset rules and signal race rule as shown in the example of the program code (having variety of design rules including constraints) starting on the column 20 and continued on the column 21;

claims 15, 30, 34: outputting a violation report, the report containing the identified violations (col. 8, ll.14-17) and as shown in the example of the program code on the columns 11-12:

```

    if both trees are valid
    {
        compare the error buffers and report each distinct
        error, as well as all duplicated errors.
    }
    else report all errors to the user

```

claim 29: a storage medium that stores the extracted information such that it is available to a set of predefined rules within a computer hardware suitable for implementing the verifying the compliance of an initial logic design against the set of design rules applicable to the type of programmable device (col. 2, ll.34-37) as shown on the Fig. 2 (col. 3, ll.6-9).

Conclusion

THIS ACTION IS MADE FINAL. Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action.

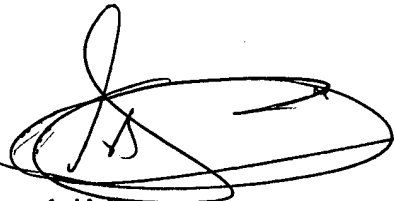
Any inquiry concerning this communication or earlier communications from the examiner should be directed to Helen Rossoshek whose telephone number is 571-272-1905. The examiner can normally be reached on 7:30-4:30.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Jack Chiang can be reached on 571-272-7483. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Art Unit: 2825

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see <http://pair-direct.uspto.gov>. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

Examiner
Helen Rossoshek
AU 2825



A. M. Thompson
Primary Examiner
Technology Center 2800